移动云计算中能效感知的计算卸载机制研究 *

杨　凡，任　丹

(湖北文理学院 计算机工程学院, 湖北 襄阳 441053)

摘　要：移动云计算可以将任务从移动设备计算卸载至云端以增强设备计算能力，而如何实现能效计算卸载机制是当前的主要挑战。为了解决该问题，以降低移动设备能耗和应用完成时间为目标，将计算卸载问题形式化为满足任务顺序与截止时间约束的能效代价最小化问题，并提出一种动态能效感知计算卸载算法。算法由三个子算法组成：计算卸载选择、时钟频率控制及传输功率分配。实验结果表明，通过局部计算时优化调整移动设备 CPU 时钟频率，以及云端计算时自适应分配传输功率，新算法可以有效降低应用执行能效代价，同时确保满足约束条件，提高执行效率。

关键词：移动云计算；计算卸载；能效代价；无线信道

中图分类号：TP302　　doi: 10.3969/j.issn.1001-3695.2018.02.0085

Energy-aware computation offloading mechanism in mobile cloud computing

Yang Fan, Ren Dan

(*School of Computer Engineering Hubei University of Arts & Science*, *Xiangyang Hubei* 441053, *China*)

Abstract: Mobile cloud computing can significantly enhance computation capability of mobile devices by offloading computation from the mobile devices onto the cloud. How to achieve energy-efficient computation offloading remains a challenge issue. For solving this problem, with reducing energy consumption and shorting application completion time as an objective, this algorithm was proposed. The algorithm will formulate computation offloading problem into the energy-eficiency cost minimization problem while satisfying the task-dependency requirements and the completion deadline constraint. And, a dynamic and energy-aware computation offloading algorithm is presented. The algorithm consists of three sub-algorithm of computation offloading selection, clock frequency control and transimission power allocation. Experimental results show that the algorithm can effective reduce the energy-eficiency cost by optimally adjusting the CPU clock frequency of mobile devices in local computing, and adapting the transmission power in cloud computing while ensures to satisfy the constraints and enhance the scheduling efficiency.

Key words: mobile cloud computing; computation offloading; energy-efficient cost; wireless channel

0　引言

目前，以智能手机、笔记本电脑为主的智能移动设备由于其便携性、简洁性得到了越来越广泛的应用，这类移动设备也将成为支持计算密集型应用的主要平台类型，服务于交互式游戏、图形图像处理、电子商务及在线社交网络服务等在内的众多领域[1]。技术层面上而言，以上领域主要涉及复杂应用类型，需要移动设备上具有高性能计算能力、内存和长周期的电池容量[2]。然而，由于物理尺寸的限制，移动设备通常是资源受限的。尤其是，电池有限的电源供给是移动设备面临的最大挑战[3]。

随着 3G、4G 和 Wi-Fi 等无线通信技术的发展，移动云计算 MCC（Mobile Cloud Computing）已经成为解决以上问题的有效手段[4]。移动云计算可以将资源丰富且计算能力更强的云扩展至资源受限的移动设备端以增强移动设备的处理能力。为了实现该目标，MCC 需要将移动设备端的资源密集型计算通过无线访问方式迁移至云端，即所谓的计算卸载（computation offloading）。MCC 中的移动应用需要分割为可执行于移动设备上的任务序列，此时称为局部执行（local execution）；执行于云端时则称为云端执行（cloud execution）。移动云计算可以选择性地通过应用的计算卸载至云端的方式提高移动设备的性能，并且节省移动设备的能耗，延长其工作时间。

尽管基于计算卸载技术的 MCC 可以较好的提高移动设备端的能力，但仍有诸多问题有待解决，而主要问题之一是如何实现高能效的计算卸载。为了实现 MCC 中移动设备的能耗节省和应用执行的高性能目标，本文重点解决了以下问题：a）哪

收稿日期：2018-02-08；**修回日期：**2018-04-10　**基金项目：**湖北省襄阳市科技计划资助项目（2015zd25）；国家自然科学基金资助项目（61272296）

作者简介：杨凡（1981-），男，湖北襄阳人，讲师，硕士，主要研究方向为数据挖掘和云计算（yangfan99_2000@163.com）；任丹（1976-），女，湖北襄阳人，讲师，硕士，主要研究方向为数据挖掘和云计算.

些应用任务需要卸载至云端计算？b)在局部移动设备上执行任务时如何决定 CPU 运行时钟频率，实现能耗优化？c）卸载任务至云端时移动设备端如何分配传输功率，以进一步优化能耗？

1　相关研究

移动计算环境中的计算卸载问题已经存在很多研究，目前涉及算法主要分为两种类型：基于性能保证的计算卸载算法[5~9]；基于能量优化的计算卸载算法[10~13]。

基于性能保证的计算卸载算法目标是提高移动设备的性能，如应用任务完成时或利用云资源的吞吐量等。此时，以资源密集为主的应用任务需要卸载至云端执行。如文献[5]提出使用虚拟机模型运行可信和资源密集型应用的调度算法，文献[6]提出 CloneCloud 动态卸载安卓代码至云端执行，文献[7]利用细粒度方式进行应用分割后进行计算卸载，以及文献[7,8]设计多用户计算分割以优化数据流的方式降低任务完成时间。然而，以上工作没有考虑能效的优化，也没有考虑移动设备上执行的应用任务间的顺序依赖对计算卸载策略的影响。

另一方面，基于能量优化的计算卸载算法目标是降低移动设备的能耗。此时，通过计算卸载降低任务计算开销可以实现该目标，因此，计算密集型应用需要计算卸载于云端执行。如文献[10]中设计了一种最小化能耗的任务调度算法，文献[11]设计的基于随机无线信道的能效最优移动云计算调度框架，文献[12]针对移动任务的协作式任务执行框架，以及文献[13]基于 Laypunov 优化法设计的动态计算卸载能耗优化算法。然而，以上工作并未做到真正的能耗优化，如局部调度时的 CPU 时钟频率控制，以及计算卸载至云端时的传输功率分配问题。文献[14]虽考虑了能耗与应用完成时间的同步优化，但所涉及的是独立任务集类型，任务间不存在顺序依赖，且未考虑计算卸载决策时的资源分配。同样地，文献[15]虽同时进行双目标优化，但在局部计算时未优化时钟频率，云端计算时且优化传输功率分配，这同样会导致能耗的增加。

然而，以上工作并没有同时考虑应用完成时间和移动设备能耗两个因素，本文将提出一种动态的计算卸载算法，在应用完成时间和任务执行顺序依赖约束下以最小化能耗和完成时间为目标，实现移动云计算环境中的任务调度优化。算法分三步进行：应用任务的计算卸载选择；局部执行时移动设备 CPU 的运行频率调整；计算卸载时传输功率分配。

2　模型描述

2.1　系统模型

假设区域内有 N 个移动设备，标识为集合 $N=\{1,2,..,N\}$，每个移动设备均拥有计算密集型应用需要执行。MCC 中的移动应用可分割为顺序的 M 个任务，表示为集合 $M=\{1,2,…,M\}$。通常，移动设备可以通过两种类型的通信方式将计算应用卸载至云端，即：移动网络或访问点方式，如图 1 所示。

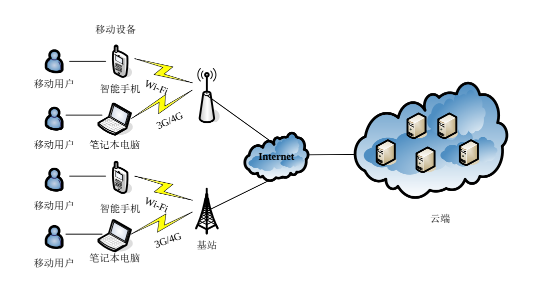


图 1 移动云计算环境

利用如图 2 的有向无循环任务图 $G=(V,E)$描述任务间的顺序关系。图 G 中的每个节点 $i\in V$ 表示一个任务，有向边 $e(i,j)$ 表示任务 i 与任务 j 间的顺序约束，即：任务 j 不能在其前驱任务 i 完成前开始执行。

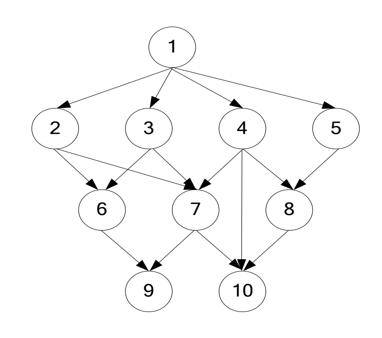


图 2 应用任务结构图

2.2　通信模型

移动云计算中通信方式以无线访问方式为主，无线访问点可以是 Wi-Fi 访问点或蜂窝网络中的基站。移动设备与无线访问点间的信道服从静态衰减模型。令 $a_{m,n}$□{0,1}表示移动设备 n 的任务 m 的计算卸载决策，$a_{m,n}=1$ 表明移动设备 n 选择通过无线信道将任务 m 计算卸载至云端，$a_{m,n}=0$ 表明移动设备 n 选择在其局部设备上执行任务 m。令所有移动设备的所有任务的决策表示为矩阵 $A=\{a_{m,n}|m$□M,n□$N\}$。对于一个给定的向量 A，可以计算移动设备 n 的任务 m 进行计算卸载时的上传数据速率为

$$R_{m,n}(A)=W\log_2\left(1+\frac{P_{m,n}^T H_{m,n}}{\sigma_{m,n}^2+\sum_{i\neq m,j\neq n,a_{i,j}=1}P_{i,j}^T H_{i,j}}\right) \quad (1)$$

其中:$P_{m,n}^T$为移动设备 n 通过无线信道卸载任务 m 的传输功率，$H_{m,n}$ 为由于路径衰减原因传输任务 m 时的信道增益，$\sigma_{m,n}^2$为与传输信道链路相关的热噪声功耗，W 为信道带宽。由式(1)可知，若多个移动设备同步通过无线访问信道进行计算卸载，将会导致严重的干扰和数据传输速率的下降。

2.3　计算模型

令 $D_{m,n}$ 为移动设备 n 的任务 m 的计算输入数据量，$L_{m,n}$ 为计算负载，表示为完成任务 m 需要的 CPU 周期总量。令 $FT_{k,n}^l$、$FT_{k,n}^t$、$FT_{k,n}^c$、$FT_{k,n}^r$分别表示移动设备 n 的任务 m 的局部计算完成时间、无线传输时间（即任务被卸载至云端执行）、云端计算时间和无线接收数据时间。以下讨论局部计算和云端计算时应用的能耗开销和完成时间开销。

1）　局部计算

令 $f_{m,n}$ 为移动设备 n 的任务 m 在局部执行时移动设备的计算能力，以 CPU 的时钟频率表示。不同移动设备可拥有不同的计算能力，且一个移动设备的不同任务可执行于不同的 CPU 频率上。则局部计算时移动设备 n 的任务 m 的执行时间为

$$T_{m,n}^{l}=\frac{L_{m,n}}{f_{m,n}} \tag{2}$$

移动设备的能耗为

$$E_{m,n}^{l}=\kappa L_{m,n}f_{m,n}^{2} \tag{3}$$

其中：κ 表示与芯片相关的电容切换因子，通常设置为 $\kappa=10^{-12}$。可以看出，调整 CPU 的频率可以实现执行时间和能耗的优化。

一个任务被执行前，其所有直接前驱任务必须已经完成。以下给出任务就绪时间的概念。

定义 1　就绪时间。一个任务的就绪时间定义为其所有直接前驱任务完成的最早时间。对于局部执的移动设备 n 的任务 m 的就绪时间则为

$$RT_{m,n}^{l}=\max_{k\in pred(m)}\max\{FT_{k,n}^{l},FT_{k,n}^{r}\} \tag{4}$$

其中：$pred(m)$为任务 m 的直接前驱任务集。

由于对于多种应用类型而言，其输出数据量通常远小于输入数据量，因此，本文忽略从云端返回数据至移动设备时的传输时间和能耗。基于该假设，式(4)可重写为

$$RT_{m,n}^{l}\geq(1-a_{k,n})FT_{k,n}^{l}+a_{k,n}FT_{k,n}^{c},k\in pred(m) \tag{5}$$

上式表明，若不考虑从无线信道接收任务 k 的结果的时间，一旦任务 k 完成，任务 m 可立即开始执行。

显然，移动设备 n 的任务 m 在局部执行时的完成时间为局部执行时间与局部计算时的就绪时间之和，即

$$FT_{m,n}^{l}=T_{m,n}^{l}+RT_{m,n}^{l} \tag{6}$$

根据式(2)和(3)，可以计算能效代价 EEC。

定义 2　能效代价 EEC。EEC 定义为执行任务的能耗与完成时间的权重之和。因此，移动设备 n 的任务 m 在局部执行时的 EEC 为

$$Z_{m,n}^{l}=\gamma_{m,n}^{E}E_{m,n}^{l}+\gamma_{m,n}^{T}FT_{m,n}^{l} \tag{7}$$

其中：$0\leq\gamma_{m,n}^{E}\leq1$，$0\leq\gamma_{m,n}^{T}\leq1$，分别表示任务 m 在计算能耗与完成时间上的权重因子。

为了满足用户需求，允许不同移动设备选择不同的权重因子进行决策。例如：对于拥有较低电池能量的移动设备可选择更大的 $\gamma_{m,n}^{E}$，以便在决策制定时节省更多能量；而当移动设备运行时延敏感型应用时（如在线电影、音乐等），可选择更大的 $\gamma_{m,n}^{T}$ 以降低时延。

2）云端计算

对于云端计算，移动设备 n 将卸载其计算任务 m 至云端执行，云端执行计算任务后返回结果至移动设备 n。任务 m 在云端执行过程包括三个阶段：传输阶段、云端计算阶段和接收阶段。

根据 2.2 节描述的通信模型，可以计算移动设备 n 卸载任务 m 时的传输时间和能耗分别为

$$T_{m,n}^{c,trs}(A)=\frac{D_{m,n}}{R_{m,n}(A)} \tag{8}$$

$$E_{m,n}^{c,trs}(A)=P_{m,n}^{T}\cdot T_{m,n}^{c,trs}(A) \tag{9}$$

进一步，可以计算任务 m 在云端的执行时间为

$$T_{m,n}^{c,exe}=\frac{L_{m,n}}{f_{c}} \tag{10}$$

其中：f_c 为云端处理单元的时钟频率，且假设 f_c 是固定的，计算期间不发生改变。

由于云端通常拥有足够的能量执行卸载任务，因此，本文不考虑任务在云端的计算能耗。

考虑到移动应用任务间通常具有依赖性，任务 m 在云端的就绪时间定义为

$$RT_{m,n}^{c}=\max\{FT_{m,n}^{t},\max_{k\in pred(m)}FT_{k,n}^{c}\} \tag{11}$$

可以看出，若任务 m 的直接前驱任务 k 在局部执行，则 $FT_{k,n}^{c}=0$。因此，$\max_{k\in pred(m)}\{FT_{k,n}^{c}\}$可视为任务 m 的所有直接前驱任务卸载至云端已经完成的时间。此外，只要任务已被完全卸载至云端或所有直接前驱任务已经在云端完成执行，任务 m 即可在云端立即执行，即

$$RT_{m,n}^{c}\geq FT_{m,n}^{t},RT_{m,n}^{c}\geq\max_{k\in pred(m)}FT_{k,n}^{c} \tag{12}$$

若忽略接收任务 m 结果的时间，移动设备 n 的任务 m 在云端的完成时间为执行时间与就绪时间之和，即

$$FT_{m,n}^{c}=T_{m,n}^{c,exe}+RT_{m,n}^{c} \tag{13}$$

因此，从式(8)~(10)可计算移动设备 n 的任务 m 在云端的 EEC 为：

$$Z_{m,n}^{c}=\gamma_{m,n}^{T}FT_{m,n}^{c}+\gamma_{m,n}^{E}E_{m,n}^{c,trs}(A) \tag{14}$$

从式(14)可以看出，移动设备 n 较低的数据传输速率 $R_{m,n}$ 将导致无线访问的高能耗和延长卸载任务于云端的时间。

3　问题形式化

给定移动设备 n 的应用任务序列集合 M，该应用的 EEC 计算为

$$Z_n=\sum_{m=1}^{M}Z_{m,n}=\sum_{m=1}^{M}(1-a_{m,n})Z_{m,n}^{l}+a_{m,n}Z_{m,n}^{c} \quad (15)$$

算法目标是选择最优计算卸载决策 A^*、CPU 时钟频率控制决策 F^*和传输功率分配决策 P^*，使得能效代价 EEC 达到最小化。综合以上模型，所有移动设备的能效计算卸载决策问题可形式化为一个约束最小化问题，即

$$\min_{A,F,P}\sum_{n=1}^{N}Z_n \quad (16)$$

约束条件如下：

$$C1:\sum_{m=1}^{M}(1-a_{m,n})FT_{m,n}^{l}+a_{m,n}(FT_{m,n}^{c})\le T_{n,\max}$$
$$C2:(1-a_{k,n})FT_{k,n}^{l}+a_{k,n}FT_{k,n}^{c}\le RT_{m,n}^{l},k\in pred(m)$$
$$C3:FT_{m,n}^{t}\le RT_{m,n}^{c}$$
$$C4:\max_{k\in pred(m)}FT_{k,n}^{c}\le RT_{m,n}^{c}$$
$$C5:a_{m,n}\in\{0,1\}$$
$$C6:\forall m\in M,n\in N$$

其中，$A=\{a_{m,n}|m□M,n□N\}$，$F=\{f_{m,n}|m□M,n□N\}$，$P=\{P^{T}_{m,n}|m□M,n□N\}$。条件 C1 给出完成时间约束，即移动设备 n 的所有任务的总完成时间需不超过截止时间 $T_{n,max}$ 的限制；条件 C2 确保任务仅能在其所有直接前驱完成后才可以开始执行，即任务顺序依赖要求；条件 C3 给出云端任务顺序约束，即确保任务仅能在完全卸载至云端后才可以开始执行；条件 C4 给出另一云端任务顺序约束，即确保任务只有在其所有直接前驱任务已经在云端完成执行后才可以开始执行；条件 C5 为计算卸载决策约束，即任务仅能在移动设备上局部执行或卸载至云端执行。

求解式(16)中带有整数约束 $a_{m,n}□\{0,1\}$的最优化问题是使其变为混合整数规划问题，而该问题通常是非凸和 NP 问题。利用松驰法，首先需要将二进制计算卸载决策变量 $a_{m,n}$ 变为 0 至 1 间的实数，即：$0\le a_{m,n}\le 1$。

以下研究拥有松驰优化变量 $a_{m,n}$ 的最优化问题(16)的凸性。

定理 1 有约束条件 C1～C6 的最优化问题(16)的凸性可对应于最优化变量$\{a_{m,n}\}$、$\{f_{m,n}\}$和$\{P^{T}_{m,n}\}$定义的优化问题。

证明 需要证明式(16)中的目标函数 $Z_{m,n}$ 是拥有优化变量$\{a_{m,n}\}$、$\{f_{m,n}\}$和$\{P^{T}_{m,n}\}$的凸函数，然后再展示 C1～C5 的凸性。由于篇幅的限制，证明略。

定理 1 表明式(16)的优化问题拥有零对偶间隔，且满足斯莱特约束条件。而零对偶间隔的结论则提供了求解式(16)对偶问题的最优求解方案。

4 算法设计

4.1 对偶问题形式化

计算卸载策略可以通过求解式(16)的对偶问题得到。首先，定义优化问题(16)的 Lagrangian 函数为 $L(\omega,\mu,A,F,P)$。Lagrangian 乘子 $\omega=[\omega_n,n=1,\ldots,N]^{\mathrm{T}}$ 对应于完成时间约束条件 C1，ω_n 表示移动设备 n 的应用总完成时间应不超过最大完成时间的代价，Lagrangian 乘子 $\mu=[\mu_{m,n},m=1,\ldots,M,n=1,\ldots,N]^{\mathrm{T}}$ 对应于云端任务顺序约束条件 C3，$\mu_{m,n}$ 表示云端任务仅在完全卸载至云端后执行的代价。

最优化问题式(16)的对偶问题则为

$$\max_{\omega,\mu}\min_{A,F,P}L(\omega,\mu,A,F,P) \quad (17)$$

式(17)的对偶问题可分解为两个层次：层次 1 即内层最小化问题，由 N 个拥有相同结构的子问题组成，可以分布式方法求解；层次 2 即外部最大化问题，是对偶问题的主问题。

以下设计分布式子算法分别求解计算卸载决策选择、时钟频率控制和传输功率分配三个子问题。

4.2 计算卸载选择

计算卸载选择子算法的目标是在满足任务顺序约束的前提下，决定哪些任务需要卸载至云端执行，从而最小化完成应用的能效代价 EEC。令 $Cost^{l}_{m,n}=Z^{l}_{m,n}+\omega_nFT^{l}_{m,n}$表示局部设备的计算代价，$Cost^{c}_{m,n}=Z^{c}_{m,n}+\omega_nFT^{c}_{m,n}$表示云端计算代价。最优计算卸载选择决策可通过求解以下最小化问题得到：

$$\max_{a_{m,n}}Cost_{m,n}^{l}+a_{m,n}(Cost_{m,n}^{c}-Cost_{m,n}^{l}) \quad (18)$$

约束条件为 C2、C4 和 C5。

显然，式(18)表示的目标函数是关于变量 $a_{m,n}$ 的线性函数，且可以观察到，若 $Cost^{c}_{m,n}\ge Cost^{l}_{m,n}$，当 $a_{m,n}□[0,1]$最小时式(18)达到最小；若 $Cost^{c}_{m,n}<Cost^{l}_{m,n}$，当 $a_{m,n}□[0,1]$最大时式(18)达到最小，即如图 3 所示。因此，计算卸载选择策略如下：

$$a_{m,n}=\begin{cases}1,if\ \ Cost_{m,n}^{c}<Cost_{m,n}^{l}\\0,otherwise\end{cases} \quad (19)$$

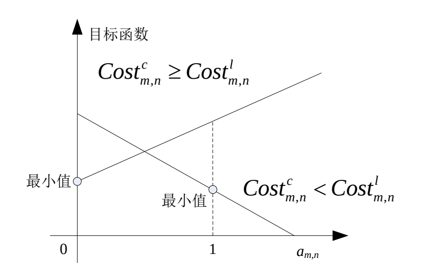


图 3 目标函数最小值

该结果表明当云端计算代价比局部计算代价更小时，任务 m 卸载至云端计算更合理。同时从 $Cost^{c}_{m,n}$和 $Cost^{l}_{m,n}$的定义可以看出，任务的计算卸载选择决策不仅取决于具体任务的时钟频率和传输功率，而且还与其直接前驱的最大完成时间和数据上传速率相关。

4.3 时钟频率控制决策

时钟频率控制决策的目标是局部计算时设置最优移动设备 CPU 的时钟频率，使应用执行的能效代价 EEC 最小。显然，当

$a_{m,n}$=0 时即任务局部执行时才会执行该策略。该策略可通过求解以下优化问题得到：

$$\min_{f_{m,n}} Z^l_{m,n} + \omega_n FT^l_{m,n} + \mu_{m,n}(FT^t_{m,n} - RT^c_{m,n}) \quad (20)$$

约束条件为 C2 和 C4。

容易证明优化问题(20)是关于 $f_{m,n}$ 的凸问题。式(20)的目标函数 $F(f_{m,n})$可重写为

$$F(f_{m,n}) = \gamma^E_{m,n}\kappa L_{m,n} f^2_{m,n} + \mu_{m,n}(FT^t_{m,n} - RT^c_{m,n}) + (\gamma^T_{m,n}+\omega_n)(L_{m,n}f^{-1}_{m,n} + RT^l_{m,n}) \quad (21)$$

由于 $RT^l_{m,n}$、$FT^t_{m,n}$和 $RT^t_{m,n}$与 $f_{m,n}$ 无关，利用标准凸优化方法和 KKT 条件，可求解时钟频率控制策略为

$$f_{m,n} = \sqrt[3]{\frac{\gamma^T_{m,n}+\omega_n}{2\kappa\gamma^E_{m,n}}} \quad (22)$$

可以看出，移动设备 n 执行任务 m 的时钟频率取决于计算完成时间权重因子、能耗权重因子以及应用总完成时间代价。

4.4 传输功率分配

传输功率分配策略的目标是优化分配移动设备在每个任务上的传输功率，使得云端计算任务时的 EEC 最小。显然，该策略仅在 $a_{m,n}$=1 即计算任务需要卸载至云端执行时才有效。传输功率分配策略可通过求解以下最小化问题得到：

$$\min_{P^T_{m,n}} \sum_{n=1}^{N}\sum_{m=1}^{M} Z^c_{m,n} + \omega_n FT^c_{m,n} + \mu_{m,n}(FT^t_{m,n} - RT^c_{m,n}) \quad (23)$$

约束条件为 C2 和 C4。

令 $Y(P^T_{m,n})$表示目标函数式(23)，根据式(8)~(14)，基于两种不同的 $RT^c_{m,n}$值可以计算两种不同形式的 $Y(P^T_{m,n})$：

情形 1 若 $FT^t_{m,n} > \max_{k\square pred(m)} FT^t_{k,n}$，即：$RT^c_{m,n}=FT^t_{m,n}$。$Y(P^T_{m,n})$时，$Y(P^T_{m,n})$可重写为：

$$Y(P^T_{m,n}) = \sum_{n=1}^{N}\sum_{m=1}^{M}[(\gamma^T_{m,n}+\omega_n)(T^{c,exe}_{m,n} + FT^t_{m,n}) + \gamma^E_{m,n}E^{c,trs}_{m,n}(A)] \quad (24)$$

其中，$FT^t_{m,n}= T^{c,trs}_{m,n}(A)+ RT^{trs}_{m,n}$。$RT^{trs}_{m,n}$表示任务 m 通过无线发送信道传输的就绪时间，为常量。

情形 2 若 $FT^t_{m,n} \le \max_{k\square pred(m)} FT^t_{k,n}$，即 $RT^t_{m,n} = \max_{k\square pred(m)} FT^t_{k,n}$ 是与 $P^T_{m,n}$无关的函数，则 $Y(P^T_{m,n})$可重写为：

$$Y(P^T_{m,n}) = \sum_{n=1}^{N}\sum_{m=1}^{M}[(\gamma^T_{m,n}+\omega_n)(T^{c,exe}_{m,n} + RT^c_{m,n}) + \gamma^E_{m,n}E^{c,trs}_{m,n}(A) + \mu_{m,n}(FT^t_{m,n} - RT^c_{m,n})] \quad (25)$$

以下先给出情形 1 中的传输功率分配策略。

利用定理 1 容易证明式(24)中的 $Y(P^T_{m,n})$是关于 $P^T_{m,n}$的凸函数。利用 KKT 条件，传输功率分配策略可由下式求解：

$$\frac{a_{m,n}}{\omega'_{m,n} + P^T_{m,n}H_{m,n}} + 1 = \ln(1 + \frac{P^T_{m,n}H_{m,n}}{\omega'_{m,n}}) \quad (26)$$

其中：$a_{m,n} = (\gamma^T_{m,n}+\omega_n)H_{m,n}/\gamma^E_{m,n} - \omega'_{m,n}$ 和 $\omega'_{m,n} = \sigma^2_{m,n} + \sum_{i\ne m, j\ne n, a_{i,j}=1} P^T_{m,n}H_{i,j}$ 分别表示热噪声功耗和从云端接收数据时的干扰。

不难看出，最优传输功率即为(26)的解，即左右两个等式的交叉点，定义左右两个式子分别为 $\varphi(P^T_{m,n})$和 $\Psi(P^T_{m,n})$，如图 4 所示。$\varphi(P^T_{m,n}) = \frac{a_{m,n}}{\omega'_{m,n} + P^T_{m,n}H_{m,n}} + 1$，$\psi(P^T_{m,n}) = \ln(1 + \frac{P^T_{m,n}H_{m,n}}{\omega'_{m,n}})$。然而，由于式(26)是一个超越方程，通常不存在对于 $P^T_{m,n}$的封闭式解，因此，仅能通过牛顿迭代法获得近似解，即传输功率 $P^T_{m,n}$通常下式迭代更新：

$$P^T_{m,n}(t+1) = P^T_{m,n}(t) - \frac{\varphi(P^T_{m,n}(t)) - \psi(P^T_{m,n}(t))}{\varphi'(P^T_{m,n}(t)) - \psi'(P^T_{m,n}(t))} \quad (27)$$

其中，$\varphi'()$和 $\Psi'()$分别为关于 $P^T_{m,n}(t)$的一阶偏导。

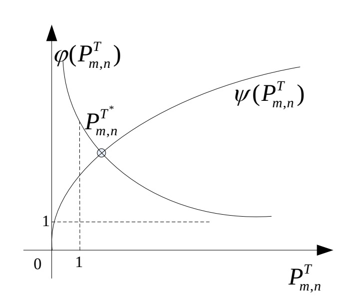


图 4 交叉点

类似情形 1，情形 2 的传输功率 $P^T_{m,n}$可迭代更新为

$$P^T_{m,n}(t+1) = P^T_{m,n}(t) - \frac{\phi(P^T_{m,n}(t)) - \psi(P^T_{m,n}(t))}{\phi'(P^T_{m,n}(t)) - \psi'(P^T_{m,n}(t))} \quad (28)$$

其中，$\phi(P^T_{m,n}(t)) = \frac{\beta_{m,n}}{\omega'_{m,n} + P^T_{m,n}H_{m,n}}$，$\beta_{m,n} = \mu_{m,n}H_{m,n}/\gamma^E_{m,n} - \omega'_{m,n}$。

从式(26)和(28)可以看出，最优传输功率与权重因子、无线信道条件和干扰是相关的。

4.5 Lagrangian 乘子更新

式(17)中层次 2 的主问题可以通过次梯度法进行求解。对于给定的 A、F、P 集合，Lagrangian 乘子可按下式进行更新：

$$\omega_n(k+1)=[\omega_n(k)+\chi(k)(T_{n,\max}-\sum_{m=1}^{M}(1-a_{m,n})FT^l_{m,n}+a_{m,n}FT^c_{m,n}]^+ \tag{29}$$

$$\mu_{m,n}(k+1)=[\mu_{m,n}(k)+\chi(k)(RT^c_{m,n}-RT^t_{m,n})]^+ \tag{30}$$

其中：k>0 为迭代次数，$\chi(k)$为正迭代步长。那么，在式(29)(30)中更新的 Lagrangian 乘子可用于更新式(19)(22)(27)(28)中的分配策略。

算法具体执行过程如算法 1 所示。算法时间复杂度为 $O(M\times Iter_{max}\times Iter_{power})$，$Iter_{max}$ 为算法的最大迭代次数，$Iter_{power}$ 为求解传输功率的牛顿迭代法的迭代次数，ε 为无穷小实数。

算法 1 能效感知的计算卸载算法

Input:$M,pred(m),Iter_{max},\varepsilon$
Output:最优分配策略$\{A,F,P\}$
Initialize $D_{m,n},L_{m,n},\gamma^E_{m,n},\gamma^T_{m,n},\chi(t),\omega_n,\mu_{m,n},\{f_{m,n}\},\{P^T_{m,n}\},t\leftarrow 1$
for m=1 to M
　while $t\le Iter_{max},|\mu_{m,n}(t+1)-\mu_{m,n}(t)|>\varepsilon$
　　/阶段 1：计算卸载选择决策/
　　compute $R_{m,n}$, $T^l_{m,n}$ and $E^l_{m,n}$ by (1)、(2) and (3)
　　if $pred(m)$==□
　　　$RT^l_{m,n}=0$, $RT^{trs}_{m,n}=0$
　　else
　　　compute $RT^l_{m,n}=\max_{k□pred(m)}\max\{FT^l_{k,n},FT^c_{k,n}\}$
　　　compute $RT^{trs}_{m,n}=\max_{k□pred(m)}\{FT^l_k\}$
　　endif
　　compute $FT^l_{m,n}$ by (6) and $Z^l_{m,n}$ by (7)
　　compute $Cost^c_{m,n}=Z^c_{m,n}+\omega_n FT^c_{m,n}$
　　compute $T^{c,trs}_{m,n}$, $E^{c,trs}_{m,n}$, $T^{c,exe}_{m,n}$ by (8)-(10)
　　compute $FT_{m,n}=T^{c,trs}_{m,n}+RT^{trs}_{m,n}$
　　if $pred(m)$ ==□
　　　$RT^c_{m,n}=T^{c,trs}_{m,n}$
　　else
　　　compute $RT^c_{m,n}$ by (11)
　　endif
　　compute $FT^c_{m,n}$, $Z^c_{m,n}$ by (13-14)
　　compute $Cost^c_{m,n}=Z^c_{m,n}+\omega_n FT^c_{m,n}$
　　if $Cost^c_{m,n}<Cost^c_{m,n}$
　　　$a_{m,n}=1$
　　else
　　　$a_{m,n}=0$
　　endif
　　if $a_{m,n}==1$
　　/阶段 2：时钟频率控制/
　　　compute the clock frequency $f_{m,n}$ by (22)
　　　$P^T_{m,n}(t+1)=P^T_{m,n}(t)$
　　else
　　/阶段 3：传输功率控制/
　　　$f_{m,n}(t+1)=f_{m,n}(t)$
　　　if $FT^c_{m,n}>\max_{k□pred(m)}FT^c_{k,n}$
　　　　compute $P^T_{m,n}$ by (22) using Newton iteration method
　　　else
　　　　compute $P^T_{m,n}$ by (24) using Newton iteration method
　　　endif
　　endif
　　/更新 Lagrangian 乘子/
　　update Lagrangian multipliers $\omega_n(t+1),\mu_{m,n}(t+1)$ by (29-30)
　　$t=t+1$
　end while
end for

5　数值仿真与性能评估

5.1　实验配置

移动云计算环境中包括 N=20 个移动智能设备，随机分布于 100m*100m 的区域中，无线访问基站位于区域的中心。对于无线访问，设置信道带宽 W=5MHz，热噪声功率 $\sigma^2_{m,n}$=50dBm。从移动设备 n 至访问点 s 的信道增益 $H_{m,n}=d^{\varsigma}_{n,s}$，$d_{n,s}$ 为两者间的距离，ς=4 为路径衰减因子。两个权重初始均设置为 $\gamma^E_{m,n}=\gamma^T_{m,n}$=0.5。迭代次数 $Iter_{max}$=200，$Iter_{power}$=100，常量 $\varepsilon=10^{-6}$。

为了评估计算卸载策略的性能，利用[8]中的计算分割方法将应用分割为 100 个任务。移动设备 CPU 最大频率配置为 2.5GHz，RAM 为 2GB，云服务器配置 4 个双核心 3.4GHz Xeon CPU，RAM 为 128G。计算任务的数据量和总的 CPU 周期数（即计算负载）服从高斯分布 $CN(\mu_1,\sigma_1^2)$和 CN(μ_2, σ_2^2)，均值 μ_1=200KB，μ_2=1000Mega 周期，标准差 σ_1=50 和 σ_2=100。分割任务数以 10 个任务作为观察目标，其任务结构和依赖性如图 2 所示。利用负载输入数据率 LDR 描述任务的复杂性，定义 $LDR_{m,n}=L_{m,n}/D_{m,n}$。将 10 个任务的输入数据 LDRs 设置为[11.056,5.499,11.35,3.011,9.522,4.742,5.052,5.786,3.676,3.925]进行实验仿真。

5.2　实验结果

1）收敛性

本实验观察任务复杂性对算法收敛性的影响。图 5 显示了任务 2～5 得到的 EEC 的收敛性情况。选择任务 2～5 是由于它们拥有相同的直接前驱任务 1。可以看出：a、算法在所有任务上在 100 次迭代内可以完成收敛；b、拥有越大 LDR 的任务收敛速度越慢，即任务复杂性将增加收敛时间；c、拥有越大 LDR 将拥有更大能效代价 EEC。因此，对任务进行计算分割时，应选择合适的 LDR 改进能效代价和降低执行时延。

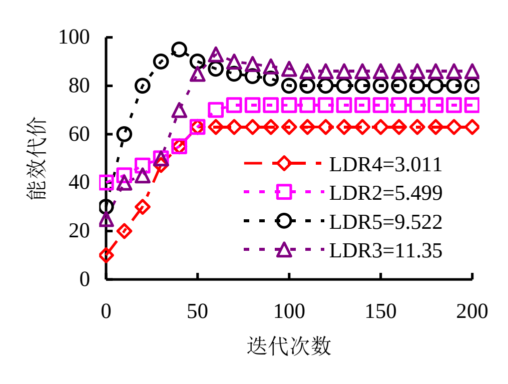
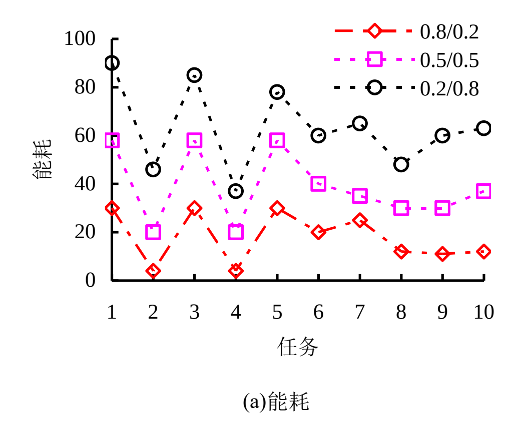


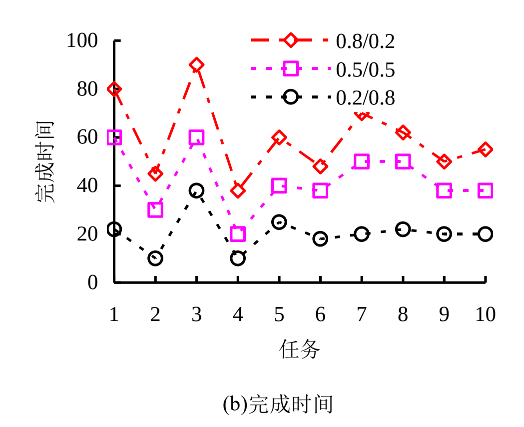
图 5 不同 LDR 下能效代价变化

2） 权重因子 $\gamma_{m,n}^{E}$和 $\gamma_{m,n}^{T}$对性能的影响

本实验观察不同前驱任务数量下两个权重因子对能耗和计算完成时间的影响，结果如图 6 所示，实验测试了权重因子 $\gamma_{m,n}^{E}$和 $\gamma_{m,n}^{T}$分别为(0.8,0.2)、(0.5,0.5)和(0.2,0.8)三种情况的结果。可以看出，对于给定的任务，能耗会随着能耗因子 $\gamma_{m,n}^{E}$的增加而降低，而对于完成时间则结果是相反的，这是由于能耗因子 $\gamma_{m,n}^{E}$的增加会导致 $a_{m,n}$ 和 $\varphi(P_{m,n}^{T})$的增加，进而会导致云端执行时传输功率的降低。同时，对于拥有相同数量直接前驱的任务，前驱任务的 LDR 值对完成时间的影响比能耗更大。且更多的直接前驱会导致轻微的能耗增加和完成时间的增加，如任务 9 和 10。



(a)能耗



(b)完成时间

图 6 权重因子能性能的影响

3） 能效代价 EEC

本实验比较算法与另外两种算法，即局部执行算法和云端执行算法，同时带有完成时间约束，进行性能比较。基准算法 1 为局部调度算法，基准算法 2 为云端调度算法。局部调度算法将所有任务调度至移动设备上执行，不进行计算卸载。云端调度算法则将所有任务均卸载至云端执行。引入这两种基准算法可以观察计算卸载对能效代价的影响。图 7 是三种算法在能效代价上的结果。可以看出，比较局部调度算法，新算法能够极大降低能效代价，当趋于稳定时新算法几乎可以降低 4 倍能效代价。这是由于新算法能够优化地选择任务在局部设备或进行计算卸载至云端执行。另外，比较云端调度算法，新算法也拥有更低的能效代价。当完成时间约束较低时，云端调度算法无法应用。随着完成时间约束变长，云端调度算法变得可用，且其能效代价出现轻微下降。然而，新算法的能效代价仍低于云端调度算法约两倍。这证明新算法中时钟频率控制和传输功率分配机制是有效可行的。

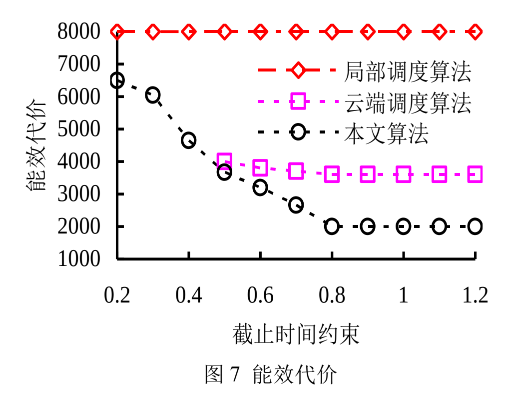
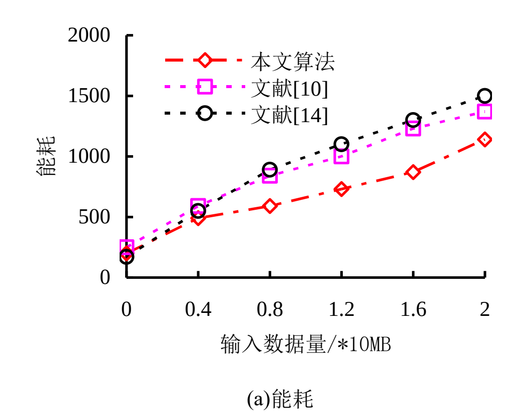


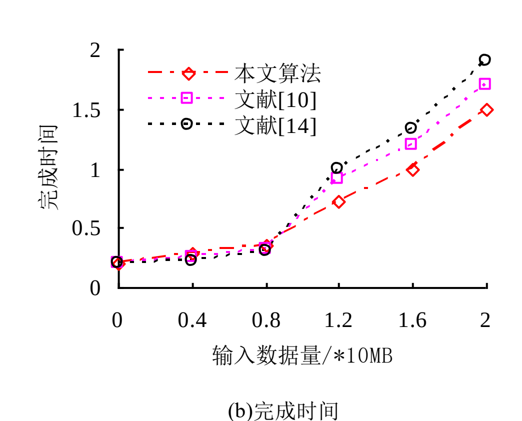
图 7 能效代价

4） 能耗与完成时间

本实验比较新算法、文献[10]和文献[14]算法在能耗和任务完成时间上的性能，结果如图 8 所示。可以看出，当输入数据量较少时，文献[14]算法拥有最少的能耗。然而，随着输入数据量的增加，能耗会快速增加，这是由于该算法拥有自适应的能耗控制机制。因此，对于大规模输入数据，新算法和算法 8 能够通过自适应调整时钟频率和传输功率降低能耗。同时，新算法比较文献[10]拥有更低的能耗，这是由于新算法不仅利用了动态频率/电压调整机制控制局部调度时的 CPU 时钟频率，而且还利用了传输功率分配机制降低了云端执行能耗。另外，从图中还可以看出，新算法的任务完成时间在输入数据量较大时增加也比较缓慢。



(a)能耗



(b)完成时间

图 8　任务规模对性能的影响

6　结束语

提出了一种移动云计算中基于能效感知的计算卸载算法，算法可以在满足应用完成时间截止限制和任务顺序依赖约束的条件下联合最小化计算能耗和完成时间，实现能效感知的计算卸载与资源调度。算法具体由三个子算法构成：计算卸载选择、时钟频率控制及传输功率分配。实验结果证实了算法在降低能耗与提高任务调度效率方面的有效性。进一步的研究可考虑基于移动设备移动性的计算卸载策略，研究移动模型对策略的影响，并进一步降低能效代价。

参考文献：

[1] Muraleedharan R. Cloud-vision: real-time face recognition using a mobile-cloudlet-cloud acceleration architecture [C]// Proc of IEEE Symposium on Computers and Communications. 2012: 59-66.

[2] Shiraz M, Gani A, Khokhar R H, *et al.* A review on distributed application processing frameworks in smart mobile devices for mobile cloud computing [J]. IEEE Communications Surveys & Tutorials, 2013, 15 (3): 1294-1313.

[3] Ma Cui, Yang Yang. A battery-aware scheme for routing in wireless Ad hoc networks [J]. IEEE Trans on Vehicular Technology, 2011, 60 (8): 3919-3932.

[4] 李继蕊，李小勇，高云全，等, 5G 网络下移动云计算节能措施研究 [J] , 计算机学报, 2017, 40 (7): 1491-1516. (Li Jirui, Li Xiaoyong, Gao Yunquan, *et al.* Energy saving research on mobile cloud computing in 5G [J] , Chinese Journal of Computers, 2017, 40 (7): 1491-1516.)

[5] Yang S, Kwon D, Yi H, *et al.* Techniques to minimize state transfer costs for dynamic execution offloading in mobile cloud computing [J]. IEEE Trans on Mobile Computing, 2014, 13 (11): 2648-2660.

[6] Chun B G, Ihm S, Maniatis P, *et al.* CloneCloud: elastic execution between mobile device and cloud [C]// Proc of International Conference on Computer Systems. New York: ACM Press, 2011: 301-314.

[7] Yang Liu, Cao Jin, Tang Sao, *et al.* A framework for partitioning and execution of data stream applications in mobile cloud computing [J]. ACM SIGMETRICS Performance Evaluation Review, 2013, 40 (4): 23-32.

[8] Yang Lei, Cao Jiannong, Cheng Hui, *et al.* Multi-user computation partitioning for latency sensitive mobile cloud applications [J]. IEEE Trans on Computers, 2015, 64 (8): 2253-2266.

[9] Kaewpuang R, Niyato D, Wang Ping, *et al.* A framework for cooperative resource management in mobile cloud computing [J]. IEEE Journal on Selected Areas in Communications, 2013, 31 (12): 2685-2700.

[10] Lin Xing, Wang Yao, Xie Qin, *et al.* Task scheduling with dynamic voltage and frequency scaling for energy minimization in the mobile cloud computing environment [J]. IEEE Trans on Services Computing, 2015, 8 (2): 175-186.

[11] Zhang Wan, Wen Yan, Guan Ken, *et al.* Energy-optimal mobile cloud computing under stochastic wireless channel [J]. IEEE Trans on Wireless Communications, 2013, 12 (9): 4569-4581.

[12] Zhang Wen, Wen Yin, Wu Du. Collaborative task execution in mobile cloud computing under a stochastic wireless channel [J]. IEEE Trans on Wireless Communications, 2015, 14 (1): 81-93.

[13] Huang Dong, Wang Ping, Niyato D. A dynamic offloading algorithm for mobile computing [J]. IEEE Trans on Wireless Communications, 2012, 11 (6): 1991-1995.

[14] Chen Xue. Decentralized Computation offloading game for mobile cloud computing [J]. IEEE Trans on Parallel & Distributed Systems, 2014, 26 (4): 974-983.

[15] 胡海洋，刘润华，胡华. 移动云计算环境下任务调度的多目标优化方法 [J]. 计算机研究与发展, 2017, 54 (9): 1909-1919. (Hu Haiyang, Liu Runhua, Hu Hua. Multi-objective optimization for task scheduling in mobile cloud computing [J] , Journal of Computer Research and Development, 2017, 54 (9): 1909-1919.)